## **CLAIMS**

## What is claimed is:

5

7

1. A method of creating run time executable code, comprising:

partitioning a processing element array into a plurality of hardware

3 \accelerators;

decomposing a program description into a plurality of kernel sections;

mapping said kernel sections into a plurality of hardware dependent

designs;\and

forming a matrix describing said hardware accelerators and said designs

configured to support run time execution.

1 2. The method of claim 1, wherein said partitioning includes partitioning

2 into digital signal processors.

- 1 3. The method of claim 1, wherein said partitioning includes partitioning
- 2 into bins.
- 1 4. The method of claim 1, wherein said mapping includes mapping into
- 2 multiple hardware contexts.

- 1 5. The method of claim 4, wherein said mapping into multiple hardware
- 2 contexts includes mapping a first set of variants.
- 1 6. The method of claim 5, wherein said first set of variants are produced
- 2 based upon resource usage.
- 1 7. The method of claim\5, wherein said mapping includes mapping a second
- 2 set of variants of said designs\configured to support multiple hardware
- 3 configurations of one of a plurality of bins.
- 1 8. The method of claim 1, wherein said mapping is performed by a place
- 2 and route.
- 1 9. The method of claim 1, wherein said decomposing is performed
- 2 manually.
- 1 10. The method of claim 1, wherein said decomposing is performed by a
- 2 software profiler.
- 1 11. The method of claim 10, wherein said decomposing includes executing
- 2 code compiled from said program description and monitoring timing of said
- 3 executing.



- 12. The method of claim 11, wherein said executing utilizes a set of test data.
- 1 13. The method of claim 11, wherein said monitoring includes determining
- 2 functions that consume a significant portion of said timing of said executing.
- 1 14. The method of claim 10, wherein said decomposing includes identifying
- 2 kernel sections by identifying regular structures.
- 1 15. The method of claim 10, wherein said decomposing includes identifying
- 2 kernel sections by identifying sections with a limited number of inputs and
- 3 outputs.
- 1 16. The method of claim 10, wherein said decomposing includes identifying
- 2 kernel sections by identifying sections with a limited number of branches.
- 1 17. The method of claim 10, wherein decomposing identifies overhead
- 2 sections.
- 1 18. The method of claim 1, wherein mapping includes creating microcode.
- 1 19. The method of claim 1, wherein said mapping includes creating context
- 2 dependent configurations.

- 20. The method of claim 1, wherein said matrix is sparsely-populated.
- 1 21. The method of claim 1, wherein said matrix is fully-populated.
- 1 22. A system for creating run time executable code, comprising:
- 2 a plurality of hardware accelerators partitioned from a processing
- 3 element array;
- a plurality of kernel sections created from a program description;
- 5 a plurality of hardware dependent designs derived from said kernel
- 6 sections; and
  - a matrix describing said hardware accelerators and said designs
- 8 configured to support run time execution.
- 1 23. The system of claim 22, wherein said hardware accelerators includes
- 2 digital signal processors.
- 1 24. The system of claim 22, wherein said hardware accelerators includes
- 2 bins.
- 1 25. The system of claim 24, wherein said bins support multiple hardware
- 2 contexts.



- 26. The system of craim 25, wherein said bins support a first set of variants configured to support said multiple hardware contexts.
- 1 27. The system of claim 26, wherein said first set of variants are produced
- 2 based upon resource usage.
- 1 28. The system of claim 27, wherein a second set of variants of said designs
- 2 are configured to support multiple hardware configurations of one of said
- 3 plurality of bins.
- 1 29. The system of claim 22, wherein said mapping is performed by a place
- 2 and route.
- 1 30. The system of claim 22, wherein said decomposing is performed
- 2 manually.
- 1 31. The system of claim 22, wherein said decomposing is performed by a
- 2 software profiler.
- 1 32. The system of claim 31, wherein said software profiler executes code
- 2 compiled from said program description, and monitors time consumed.

John

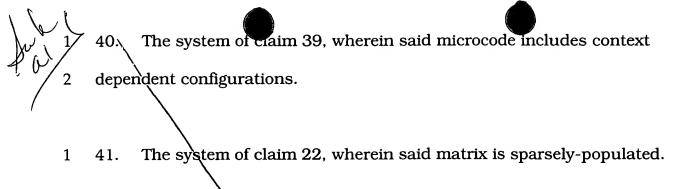
- 33. The system of craim 32, wherein said software profiler includes a set of
- 2 test data
- 1 34. The system of claim 32, wherein said software profiler determines
- 2 functions that consume a significant portion of said time consumed.
- 1 35. The system of claim 31, wherein said software profiler is configured to
- 2 identify kernel sections by identifying regular structures.
- 1 36. The system of claim 31\, wherein said software profiler is configured to
- 2 identify kernel sections by identifying sections with a limited number of inputs
- 3 and outputs.
- 1 37. The system of claim 31, wherein said software profiler is configured to
- 2 identify kernel sections by identifying sections with a limited number of
- 3 branches.
- 1 38. The system of claim 31, wherein said profiler identifies overhead
- 2 sections.
- 1 39. The system of claim 22, wherein said designs include microcode.

5

6

7

8



- 1 42. The system of claim 22, wherein said matrix is fully-populated.
- 1 43. A machine-readable medium having stored thereon instructions for 2 processing elements, which when executed by said processing elements 3 perform the following:
  - partitioning a processing element array into a plurality of hardware accelerators;
  - decomposing a program description into a plurality of kernel sections;
    mapping said kernel sections into a plurality of hardware dependent
    designs; and
- 9 forming a matrix describing said hardware accelerators and said designs 10 configured to support run time execution.

44.	A system configured to create run time executable code, comprising:
2	means for partitioning a processing element array into a plurality of
3	hardware accelerators;
4	means for decomposing a program description into a plurality of kernel
5	sections;
6	means for mapping said kernel sections into a plurality of hardware
7	dependent designs; and
8	means for forming a matrix describing said hardware accelerators and
9	said designs configured to support run time execution.